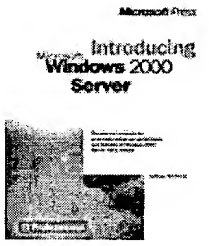
Active Directory

Chapter 11 from *Introducing Windows 2000 Server*, published by Microsoft Press

On This Page



Overview

Keeping track of everything on your network is a time-consuming task. Even on small networks, users tend to have difficulty finding network file and printer shares. Without some kind of network directory, medium and large networks are *impossible* to manage, and users will often have a difficult time finding resources on the network.

Previous versions of Microsoft Windows included services to help users and administrators find network resources. Network Neighborhood is useful in many environments, but users often complain about the clumsy interface, and its unpredictability baffles many administrators. The WINS Manager and Server Manager could be used to view a list of systems on the network, but they were not readily available to end users. Administrators utilized User Manager to add and delete users, an entirely different type of network object. These applications got the job done, but proved to be inefficient—especially in large networks.

All of these objects resided in a common container: the Microsoft Windows NT domain. Windows NT domains worked best in small-sized and medium-sized environments. Administrators of large environments were forced to partition their network into multiple domains interconnected with trusts. Microsoft Windows 2000 Server introduces Active Directory to replace domain functionality. Active Directory will continue to get the job done, but in a much more efficient way. Active Directory can be replicated between multiple domain controllers, so no single system is critical. In this way, the crucial data stored within Active Directory is both redundant and load-balanced.

A directory, in the most generic sense, is a comprehensive listing of objects. A phone book is a type of directory

that stores information about people, businesses, and government organizations. Phone books typically record names, addresses, and phone numbers. Active Directory is similar to a phone book in several ways, and it is far more flexible. Active Directory will store information about organizations, sites, systems, users, shares, and just about any other network object that you can imagine. Not all objects are as similar to each other as those stored in the phone book, so Active Directory includes the ability to record different types of information about different objects. This chapter will teach you

- What Active Directory is
- How standard protocols like DNS dynamic update protocol and Lightweight Directory Access Protocol (LDAP) are used
- How to plan for migrating to Active Directory
- What objects, schema, object classes, and attributes are
- How replication and partitioning work
- What the global catalog is useful for and how to use it

Top of page

Active Directory Components

As I mentioned in the introduction, Active Directory stores information about network components. It allows clients to find objects within its *namespace*. The term namespace (also known as *console tree*) refers to the area in which a network component can be located. For example, the table of contents of this book forms a namespace in which chapters can be resolved to page numbers. DNS is a namespace that resolves host names to IP addresses. Telephone books provide a namespace for resolving names to telephone numbers. Active Directory provides a namespace for resolving the names of network objects to the objects themselves. Active Directory can resolve a wide range of objects, including users, systems, and services on a network.

Everything that Active Directory tracks is considered an *object*. An object is any user, system, resource, or service tracked within Active Directory. The generic term *object* is used because Active Directory is capable of tracking a variety of items, and many objects can share common *attributes*.

Attributes describe objects in Active Directory. For example, all User objects share attributes to store a user name, full name, and description. Systems are also objects, but they have a separate set of attributes that include a host name, an IP address, and a location.

The set of attributes available for any particular object type is called a *schema*. The schema makes object classes different from each other. Schema information is actually stored within Active Directory, which allows administrators to add attributes to object classes and have them distributed across the network to all corners of the domain, without restarting any domain controllers.

A *container* is a special type of object used to organize Active Directory. It does not represent anything physical, like a user or a system. Instead, it is used to group other objects. Container objects can be nested within other containers.

Each object in an Active Directory has a *name*. These are not the names that you are accustomed to, like "Tony" or "Eric." They are LDAP *distinguished names*. LDAP distinguished names are complicated, but they allow any object within a directory to be identified uniquely regardless of its type. My distinguished name on the Microsoft network is "/O=Internet/DC=COM/DC=Microsoft/ DC=MSPress/CN=Users/CN=Tony Northrup"...but you can call me Tony.

The term *tree* is used to describe a set of objects within Active Directory. When containers and objects are combined hierarchically, they tend to form branches—hence the term. A related term is *contiguous subtree*, which refers to an unbroken branch of the tree.

Continuing the tree metaphor, the term *forest* describes trees that are not part of the same namespace but that share a common schema, configuration, and global catalog. Trees in a forest all trust each other, so objects in

these trees are available to all users if the security allows it. Organizations that are divided into multiple domains should group the trees into a single forest.

A *site* is a geographical location, as defined within Active Directory. Sites correspond to logical IP subnets, and as such, they can be used by applications to locate the closest server on a network. Using site information from Active Directory can profoundly reduce the traffic on wide area networks.

↑ Top of page

Managing Active Directory

The Active Directory Users and Computers MMC snap-in is the most useful tool for administering your Active Directory. It is directly accessible from the Administrative Tools program group on the Start menu. It replaces and improves upon Server Manager and User Manager from Windows NT 4.0. Take a few minutes to familiarize yourself with this tool. It is very intuitive—just be sure not to make any modifications until you understand how Active Directory works.

↑ Top of page

Security

Active Directory plays an important role in the future of Windows networking. Administrators must be able to protect their directory from attackers and users, while delegating tasks to other administrators where necessary. This is all possible using the Active Directory security model, which associates an access control list (ACL) with each container, object, and object attribute within the directory. Figure 11-1 shows a step from the Delegation Of Control wizard, a helpful utility for assigning permissions to Active Directory objects.

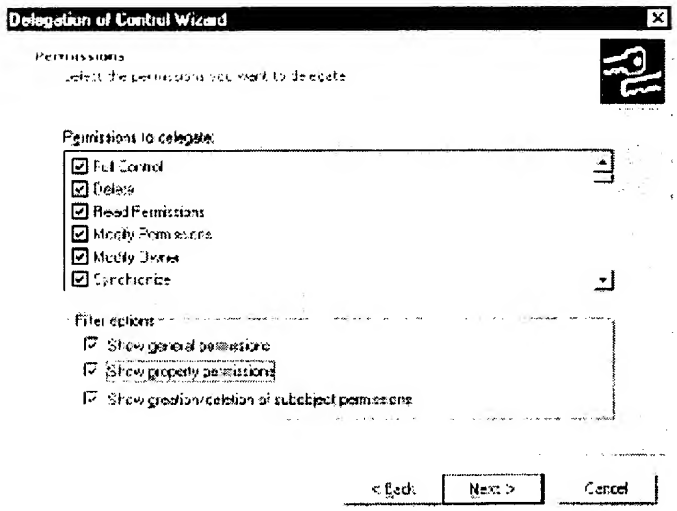


Figure 11-1: The Delegation Of Control wizard makes it simple to assign permissions to objects.
See full-sized image.

This high level of control allows an administrator to grant individual users and groups varying levels of permissions for objects and their properties. Administrators can even add attributes to objects and hide those attributes from certain groups of users. For example, the administrator could set the ACLs such that only managers can view the home phone numbers of other users. Nonmanagers would not even know that the attribute existed.

A concept new to Windows 2000 Server is *delegated administration*. This allows administrators to assign administrative tasks to other users, while not granting those users more power than necessary. Delegated administration can be assigned over specific objects or contiguous subtrees of a directory. This is a much more effective method of giving authority over the networks; rather than granting someone the all powerful Domain Administrator permissions, he or she can be given permissions for just those systems and users within a specific

subtree. Active Directory supports *inheritance*, so any new objects inherit the ACL of their container.

Try to forget what you've learned about Windows NT domain trusts. The term *trusts* is still used, but trusts have very different functionality. There is no distinction between one-way and two-way trusts because all Active Directory trusts are bidirectional. Further, all trusts are transitive. So, if Domain A trusts Domain B, and Domain B trusts Domain C, then there is an automatic implicit trust between Domain A and Domain C. This new functionality is shown in Figure 11-2.

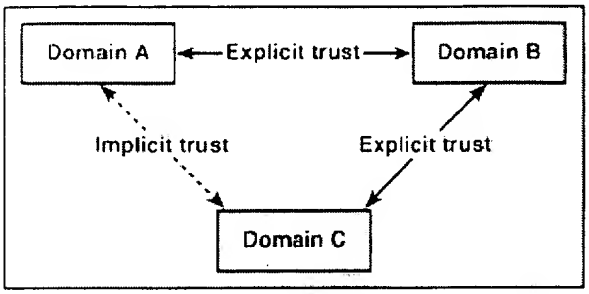


Figure 11-2: Windows 2000 Server trusts are bidirectional and transitive.

Another Active Directory security feature is auditing. Just as you can audit NTFS partitions, objects and containers within Active Directory can be audited. This is a useful way to determine who is attempting to access objects, and whether or not they succeed.

Top of page

Use of DNS (Domain Name System)

Domain Name System, or DNS, is necessary to any Internet-connected organization. DNS provides name resolution between common names, such as mspress.microsoft.com, and the raw IP addresses that network layer components use to communicate. Active Directory makes extensive use of DNS technology and relies on DNS to locate objects within Active Directory. This is a substantial change from previous Windows operating systems that require NetBIOS names to be resolved to IP addresses, and to rely on WINS or another NetBIOS name resolution technique.

Active Directory works best when used with Windows 2000-based DNS servers. Microsoft has made it easy for administrators to transition to Windows 2000-based DNS servers by providing migration wizards that walk the administrator through the process. Other DNS servers can be used, but administrators will need to spend more time managing the DNS databases. If you decide not to use Windows 2000-based DNS servers, you should make sure your DNS servers comply with the new DNS dynamic update protocol. Active Directory servers rely on dynamic update to update their pointer records, and clients rely on these records to locate domain controllers. If dynamic update is not supported, you will have to update the databases manually.

Note: DNS dynamic update protocol is defined in RFC 2136.

Windows domains and Internet domains are now completely compatible. A domain name such as mspress.microsoft.com will identify Active Directory domain controllers responsible for the domain, so any client with DNS access can locate a domain controller. Active Directory clients can use DNS resolution to locate any number of services because Active Directory servers publish a list of addresses to DNS using the new features of dynamic update. These addresses identify both the domain and the service being provided and are published via Service Resource Records (SRV RRs). SRV RRs follow this format:

service.protocol.domain

Active Directory servers provide the LDAP service for object location, and LDAP relies on TCP as the underlying transport-layer protocol. Therefore, a client searching for an Active Directory server within the mspress.microsoft.com domain would look up the DNS record for ldap.tcp.mspress.microsoft.com.

Top of page

Global Catalog

Active Directory provides a *global catalog* (GC). No, this does not mean that you can find any piece of information on the planet—but it is still very significant. Active Directory provides a single source to locate any object within an organization's network.

The global catalog is a service within Windows 2000 Server that allows users to find any objects to which they have been granted access. This functionality far surpasses that of the Find Computer application included in previous versions of Windows, because users can search for any object within Active Directory: servers, printers, users, and applications. For example, Figure 11-3 shows how a user can search for all color printers in his or her building that have the capability to print double-sided documents.

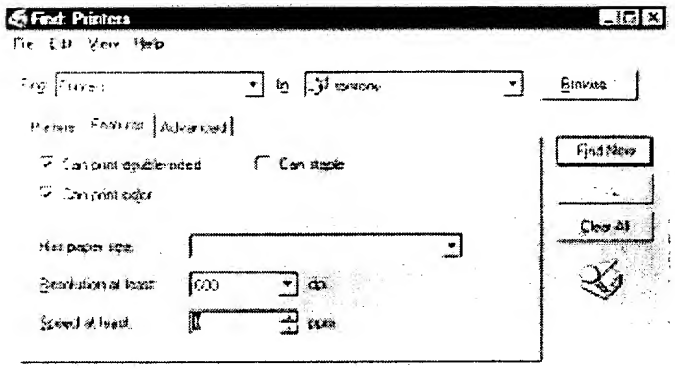


Figure 11-3: The global catalog helps users find network resources.
See full-sized image.

This feature is especially important because of the complexity of LDAP names. Older versions of Windows relied on 15-character NetBIOS computer names, which users could often remember. Few people would be able to recall LDAP names, such as the following:

```
/O=Internet/DC=COM/DC=Microsoft
/DC=MSPress/CN=Computers/CN=Server1.
```

Because users can easily search for objects, remembering names is much less important.

The GC is an index stored on Active Directory servers. It contains the names of all objects in the Active Directory server, regardless of how the server has been partitioned. The GC also contains a handful of searchable attributes for each object. For example, the GC would store the distinguished names, first names, and last names of all users—allowing someone to search for anyone named Tony and find the distinguished name of the user. The global catalog is a subset of Active Directory, and stores only those attributes that users tend to search on. Useful defaults are provided by Microsoft, and administrators can specify other attributes to be searchable by using the Active Directory Schema, described later in this chapter.

Not All Indexes Are Created Equal!

If you have done any database administration, you already know that some types of information are more useful to index than other types. Naturally, you should index attributes that will be searched for often, but there are other factors involved. Indexes take up space, so it is not efficient to index everything. Indexes also slow down updates and inserts—if an indexed attribute is modified, the index must be modified as well. Indexing works better when the data being stored varies from user to user. Therefore, never index true or false attributes or any attribute with less than five possible values. Names are an excellent attribute to index since they are almost unique for each user. Finally, don't index attributes that aren't usually filled in. If few users enter a value for their middle name, the indexing of that attribute is a waste.

As new objects are created in Active Directory, they are assigned a unique number called a GUID (globally unique identifier). The GUID is useful because it stays the same for any given object, regardless of where the object is moved. The GUID is a 128-bit identifier, which isn't particularly meaningful to users, but applications that reference objects in Active Directory can record the GUIDs for objects and use the global catalog to find them even after they've moved.

^ Top of page

Replication

Administrators who implement Active Directory will quickly discover that their network relies heavily on its services. This reliance means that Active Directory must be available on multiple servers—so that if a single server fails, clients can contact a server with duplicate services and information. Unlike the Windows NT domain databases used with previous versions of Windows NT, updates to the database can be sent to *any* of the Active Directory servers. While this complicates the replication process, it eliminates the possibility that the failure of a single domain controller would stop updates to the databases. It also reduces the high load placed on Windows NT 4.0 primary domain controllers.

Windows 2000 Server includes a replication component within the suite of Active Directory services that makes this a simple task for administrators. Simply adding domain controllers to an Active Directory domain is sufficient to begin the replication process.

One of the most complex parts of making redundant servers work properly is replicating the information and ensuring that all servers have the most up-to-date content. Active Directory uses *multimaster replication,* which is another way of stating that updates can occur on any Active Directory server. Each server keeps track of which updates it has received from which servers, and can intelligently request only necessary updates in case of a failure.

How Active Directory Replication Works

Active Directory replication will seem logical if you're already familiar with how replication works in Windows NT 4.0 domains. Each update is assigned its own 64-bit unique sequence number (USN) from a counter that is incremented whenever a change is made. These updates are system-specific, so every Active Directory server maintains a separate counter.

When a server replicates an update to other Active Directory servers, it sends the USN along with the change. Each server maintains an internal list of replication partners and the highest USN received from them. The server receiving the update requests only those changes with USNs higher than previously received. This method has the added benefit of stopping updates from propagating endlessly between multiple Active Directory servers.

One problem inherent in any multimaster replication scheme is that updates to a single object can occur in multiple places at the same time. For example, if an administrator in Boston changes a user's name from "Curt" to "Kurt" and an administrator in Chicago simultaneously changes that same user's name from "Curt" to "Kirk," a replication collision will occur. There are two problems to deal with when a collision occurs: detecting the collision and resolving the collision.

Active Directory stores *property version numbers* to allow replication collision detection. These numbers are specific to each property of every object within Active Directory and are updated every time the property is modified. These numbers are propagated through Active Directory along with the change, so a server that receives two different updates to the same property with the same property version number can conclude that a replication collision has occurred.

Active Directory servers resolve collisions by applying the update with the later timestamp. The timestamp is created by the server that initiated the change, so it is very important to keep system time synchronized between Windows 2000 servers.

Note: Use the built-in distributed time synchronization service to keep all servers working together!

^ Top of page

Partitioning

Large networks can contain hundreds of thousands of objects. Windows NT required multiple domains to allow that many objects to be manageable. Administrators often divided users and resources into separate domains and created a trust between the domains. The structure of the databases simply did not allow them to grow to hundreds of thousands of objects. These size limitations are less a factor in Active Directory domains, thankfully. However, supporting a very large Active Directory could be an incredible burden to any single domain controller.

Active Directories can be partitioned to lessen this load. Partitioning allows different domain controllers to manage different sections of the database, reducing the load on any individual server. The clients can use resources located within different Active Directory partitions transparently. Therefore, administrators can manage massive Active Directory domains without requiring domain controllers to handle the entire database.

↑ Top of page

Schema: Attributes and Object Classes

As I defined the term earlier, a schema is a set of attributes used to describe a particular object class in Active Directory. Different types of information need to be tracked for different object classes, and that's why the schema is so important. For example, the Users object class needs attributes for a first name, last name, phone number, e-mail address, and mailing address. The Printer object class must have many different attributes—users will want to know how fast a printer is and whether it can duplex or print in color. These attributes can be viewed and edited using the Active Directory Schema MMC snap-in, as shown in Figure 11-4. The Active Directory Schema does not have an icon within the Start menu; you must launch the MMC interface and add the snap-in named Active Directory Schema.

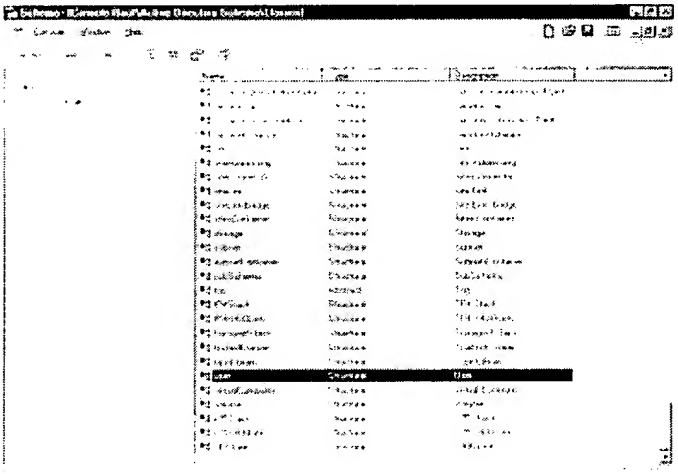


Figure 11-4: The Active Directory Schema allows classes and attributes to be modified.
See full-sized image.

By default, object classes come with a logical set of attributes that will fit most organization's needs. However, many organizations will need to track additional information about particular object classes. For example, if employees are assigned a badge number, it is useful to track that information in the object class. The first step is to create an attribute called BadgeID, as shown in Figure 11-5 on the next page. The second step is to make the new attribute optional for the Users class.

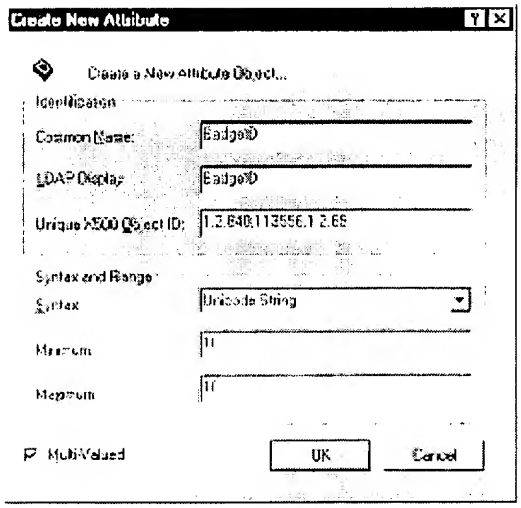


Figure 11-5: Attributes can be added with the Active Directory Schema snap-in.

The schema is stored within Active Directory just like other objects. Therefore, the schema inherits the ability to be automatically replicated throughout a domain. It also benefits from the security features of Active Directory, and allows administrators to delegate authority over the schema to different users and groups. By changing the ACLs on a schema object, an administrator can allow any user to add or modify attributes for an object class. The example in Figure 11-6 shows that the group East Coast Administrators has been granted full control over the schema.

Editing the Schema Isn't All *That* Easy!

By default, Active Directory servers do not allow the schema to be edited. Before this can be done, you must add a REG_DWORD value to the Registry named Schema Update Allowed and set it to 1. This value should be added to the following Registry key:

```
\HKLM\SYSTEM\CurrentControlSet\Services\NTDS\Parameters
```

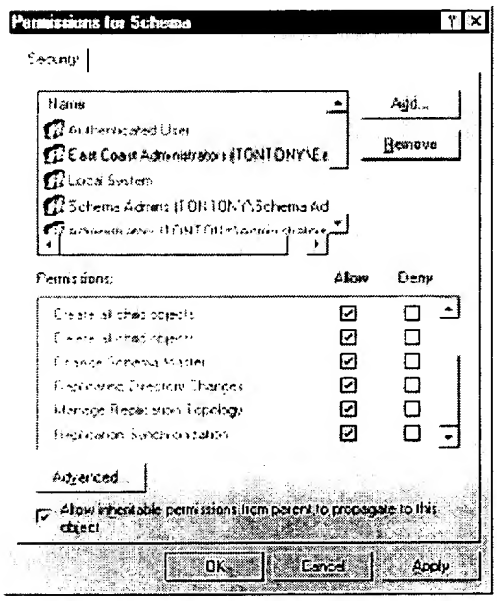


Figure 11-6: Modifying the schema can be delegated to groups and users.

New attributes have several properties that must be set. The user creating a new attribute must define a name for the attribute (such as Badge ID #), the type of data to be stored (such as a string or a number), and the range limits (such as string length). A unique Object Identifier (OID) must also be provided. New attributes can be indexed, which adds the attributes to the global catalog. Indexes should be created for attributes that users will search with. In this example, if security needs to look up user accounts by the Badge ID number, this attribute should be indexed. For a search to occur on a nonindexed attribute, a slow and processor-intensive walk of the directory tree must be done.

You cannot delete a class or an attribute with the Active Directory Schema or any other tool. Once you create them, they will exist forever within your Active Directory. The only option you have is to deactivate a class, which stops it from being used in the future. You cannot deactivate a class or an attribute that has dependencies within Active Directory. For example, if an attribute is still used by an active class, that attribute must remain active.

Where Do Object Identifiers Come From?

The only way to ensure Object Identifiers are globally unique is to have a central agency that assigns OIDs. This is already common practice on the Internet; the InterNIC assigns domain names and the Internet Assigned Numbers Authority (IANA) assigns IP subnets. Object Identifiers are assigned by a National Registration Authority, or NRA. NRAs vary from country to country. In the United States, the American National Standards Institute (ANSI) provides NRA services. For a modest fee, ANSI can supply your organization with a root OID. Any objects created by your organization will have this root OID as the prefix, ensuring that Object Identifiers are globally unique.

A list of NRAs can be found at the International Standards Organization's Web site, at http://www.iso.ch.

The schema is cached by Active Directory servers for performance reasons. It will take up to five minutes for the cache to be updated after you change the schema. So, wait a few minutes before you try to create objects based on your new object classes and attributes. If you must reload the cache immediately, add the attribute schemaUpdateNow to the root object (the object without a distinguished name), and set the value to 1.

Extending the schema of Active Directory is a powerful capability. However, most administrators will never need to use anything but the classes and attributes Microsoft has provided by default.

Top of page

Objects

Many people are initially confused by the relationship between object classes, attributes, and the objects themselves. Objects are created based on an object class. Attributes describe an object class. When an object is created, it inherits all the attributes of its object class. Here's where it gets tricky: *object classes and attributes are also objects in Active Directory.* Fortunately, most user interfaces hide this fact.

An object can be either a reference to something concrete or the actual useful information itself. For example, every bit of information about a user account is stored within Active Directory. However, only a reference to a disk volume is stored in Active Directory. While the reference is not useful by itself, it is used to locate the volume on the file server. When creating new object classes, carefully consider whether the object will store a reference to something external or whether all necessary information will be contained in the object's attributes. While Active Directory is extremely convenient, it should not be used to store large amounts of information, constantly changing information, or rarely used information.

Anytime you add a user or a computer to Active Directory, you are creating an object. Creating an object is often referred to as *publishing,* because it kicks off a process of replicating the new information across all Active Directory servers in the domain.

Top of page

Standard Object Classes

Windows 2000 Server relies on Active Directory to store a great deal of useful information about users, groups, and machine accounts, which are of particular interest to administrators because they will be the most commonly accessed parts of Active Directory. The new user interface might not seem intuitive if you're an administrator of previous versions of Windows NT, but once you spend some time with it, things will be easier.

Users

User accounts are no longer managed using a dedicated utility. Instead, administrators use the Active Directory Users and Computers MMC snap-in, as shown in Figure 11-7 on the next page. The user accounts themselves have changed significantly, as well. Windows NT 4.0 simply tracks the user name, full name, description, password, and a handful of other attributes for each user. Windows 2000 Server takes advantage of Active Directory to extend these attributes. You can now use Active Directory to track a great deal of personal information about people, including phone number, address, and manager name. All of this additional information is entirely optional.

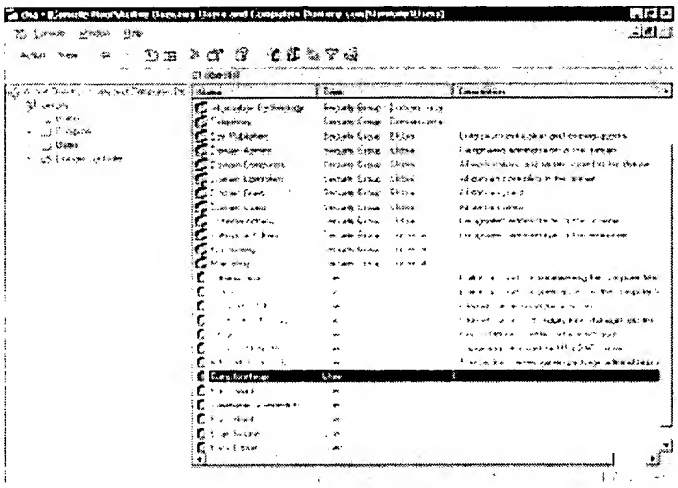


Figure 11-7: The Active Directory Users and Computers snap-in replaces the User Manager.
See full-sized image.

Groups

Active Directory groups are similar to user groups in previous versions of Windows NT. However, they have a couple of new features as well. The newest version of Microsoft Exchange allows groups to be used as e-mail distribution lists. To make this more useful, e-mail accounts can be added to the groups to allow distribution to users who are not members of the same Active Directory tree. Groups can also be nested within each other. This will greatly reduce the amount of time administrators spend managing users and groups.

There are now three distinct types of user groups. Universal Groups will be the most commonly used type, and can contain users and other groups from anywhere in the forest. They are replicated outside of the domain and appear in the global catalog. Global Groups can only contain users and groups from the same domain. Global Groups are listed in the global catalog, but their membership list does not leave the domain. Domain Local Groups can only be applied to ACLs (access control lists) within the same domain but can contain users and groups from other domains. They are neither replicated outside of the domain nor listed in the global catalog. Any of these types of groups can participate in domain security or merely function as a distribution list.

Many groups are provided by Windows 2000 Server by default. These groups are called the *built-in* groups, and are pictured in Figure 11-8. Administrators can use these default groups for most purposes, and can add their own groups as needed.

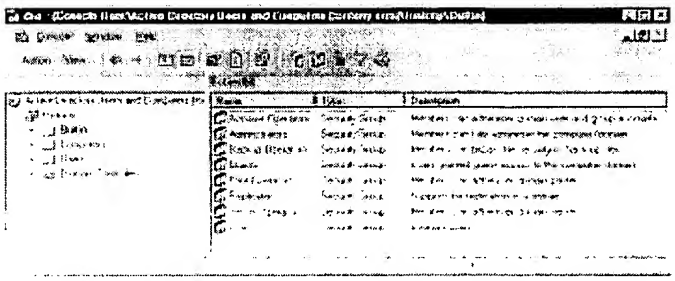


Figure 11-8: Windows 2000 Server provides many built-in groups.
See full-sized image.

Machine Accounts

Systems that join a domain are automatically given a computer account in Active Directory. This is similar to adding a system to a Windows NT 4.0 domain. However, systems can be added to the domain even if they do not participate in domain security. For example, a computer object can be created for a UNIX system to help the administrators track that system.

Top of page

Lightweight Directory Access Protocol (LDAP)

Active Directory reflects Microsoft's trend toward relying on standard protocols. The Lightweight Directory Access Protocol (LDAP) is a product of the IETF (Internet Engineering Task Force). It defines how clients and servers exchange information about a directory. LDAP version 2 and version 3 are used by Windows 2000 Server's Active Directory.

Distinguished Names

It is very important to understand the structure of distinguished names, as you will be referring to them often in the course of your job. My distinguished name is /O=Internet/DC=COM/DC=Microsoft/DC=MSPress/CN=Users/CN=Tony Northrup. Consider Figure 11-9, which shows how I fit into a sample Active Directory tree. The distinguished name I gave starts to make some sense—it identifies each container from the very top down to my specific object. Each container is separated by a slash and an identifier. For example, COM, Microsoft, and MSPress are each preceded by /DC=. The DC stands for Domain Component, which identifies a DNS domain.

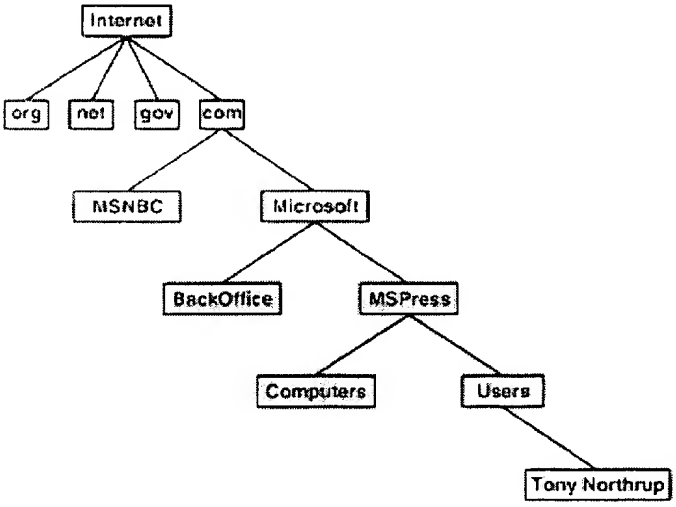


Figure 11-9: Distinguished names describe the location of an object in a tree.
See full-sized image.

To simplify distinguished names, relative distinguished names can also be used. The relative distinguished name of the previous example is CN=Tony Northrup, identifying the user name but not the context in which it resides. The context must be known already for the relative distinguished name to be an effective identifier.

User Principal Name

Distinguished names are great for computers but too cumbersome for people to remember. People have grown accustomed to e-mail addresses, so Active Directory provides these addresses as a shortcut to the full object name. In Figure 11-9, Tony Northrup is a user of the mspress.microsoft.com domain. An administrator could create a user principal name within the microsoft.com domain to allow simpler access to my user account and hold a place for my e-mail address, like northrup@microsoft.com.

Users will rely on their user principal name to log onto their Windows 2000 systems. In other words, user principal names will replace the user names used in older Windows networks. Obviously, this helps the users by saving them the trouble of typing their distinguished names. However, it also benefits users because the user principal name will stay the same even if administrators move or rename the underlying user account.

Top of page

ADSI (Active Directory Service Interface)

ADSI (Active Directory Service Interface) allows applications to interact with any directory service without being forced to know the internal details of the underlying protocols. Administrators can write programs and scripts that make use of ADSI to read or write to legacy Windows NT 4.0 directories, NetWare NDS directories, NetWare 3 binderies, and LDAP directories such as Active Directory. Developers can even create applications that make use of directories at the customer's site, without previous knowledge of the type of directory being used.

For example, the following Microsoft Visual Basic code uses ADSI to display a list of users in the debug window:

```
Set ou = GetObject("LDAP://dcserver/OU=Sales,
DC=ArcadiaBay,DC=COM")
For Each obj In ou
    Debug.Print obj.Name
Next
```

As you can see, gathering a list of users is much simpler than in previous Windows operating systems. ADSI makes use of the Component Object Model (COM), so almost any Windows development environment can immediately make use of the interface. Developers will be interested to know that they can access Active Directory through the LDAP C API and through MAPI, though ADSI is the preferred interface.

Note: The LDAP C API is defined in RFC 1823.

Top of page

Planning Your Network for Active Directory

Clients rely on site information to identify the closest Active Directory server. Because sites correspond to IP subnets, you should place Active Directory servers on each subnet. You should also make sure that all systems on the same logical subnet are connected via LAN hardware. Some routing technologies, such as Proxy Address Resolution Protocol (ARP), can allow systems to be on the same logical subnet but different physical network segments. This setup will trick clients into thinking systems are closer than they really are, so it's best to stick to standard routing techniques. If this isn't making much sense, that's okay—your network is probably set up just fine.

Make sure you have planned your Active Directory structure before you start migrating your network. You'll be given the option of creating a new tree or joining an existing tree. Obviously, if you're the first domain in the network to be migrated, you'll want to create a new tree. However, if you are merging multiple domains into a

single Active Directory domain, you will want to join as a child of the existing tree.

Always migrate the Windows NT 3.51 or 4.0 PDC (Primary Domain Controller) to Windows 2000 Server Active Directory first. Users and groups from your current domain will be automatically transferred into Active Directory, and existing clients will interface with the new domain controller exactly as if it were still a PDC. As long as you have both Active Directory servers and legacy BDCs (backup domain controllers) in operation simultaneously, your domain will function as a *mixed mode domain*, as illustrated in Figure 11-10. Mixed domains cannot take full advantage of the new Active Directory features because Active Directory must ensure backward compatibility. For example, you cannot use nested groups in mixed mode domains.

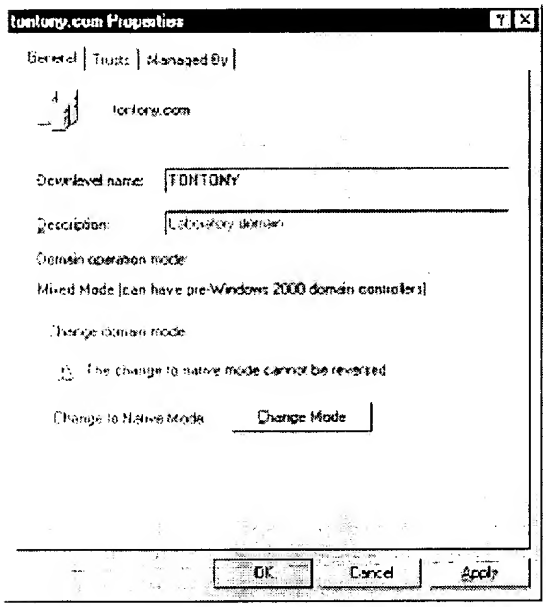


Figure 11-10: Mixed mode domains are used when legacy BDCs still exist.

You should migrate the BDCs once you are sure the mixed mode domain is functioning completely. When all domain controllers have been migrated, you can switch the domain to *native mode,* reboot the domain controllers, and take full advantage of the new features. Member servers and workstations are completely supported and require no changes to interact with Active Directory servers. You will realize more benefits by upgrading the member servers as well, but always start by upgrading domain controllers.

Windows NT Workstation clients should be upgraded to Windows 2000 Professional to take advantage of the new features of Active Directory. A service pack will be made available for Microsoft Windows 95 and Windows 98 clients that will make them Active Directory–aware and allow them to participate in Kerberos security.

Top of page

Summary

The addition of Active Directory to Windows 2000 Server is the most significant reason to upgrade your network. Active Directory combines Windows NT domains with Internet domains and makes them scalable to enterprise proportions. While the most significant benefit will be the reduced cost of ownership, users will directly benefit from the advanced search capabilities of the global catalog.

Active Directory is both standards-based and flexible. It's based on the LDAP standard, which has already been adopted by Cisco for use on network hardware and UNIX systems. The flexibility will be appreciated by any administrator who needs more functionality than is provided out of the box.

Microsoft wants it to be as easy as possible to migrate to Active Directory. Wizards are provided to transfer DNS responsibilities to Microsoft DNS dynamic update protocol servers. Users and groups from legacy Windows NT domains are automatically imported. Finally, every aspect of Active Directory setup is intuitive and GUI-

oriented, and handles most complexities automatically.

Top of page

Manage Your Profile